SOFTWARE REQUIREMENT SPECIFICATION DOCUMENT

FOR

PAST QUESTION REPOSITORY SYSTEM

Contents

[1. INTRODUCTION 3](#_Toc198552773)

[1.1 Purpose 3](#_Toc198552774)

[1.2 Scope 3](#_Toc198552775)

[1.3 Definitions, Acronyms, and Abbreviations 3](#_Toc198552776)

[1.4 Overview 4](#_Toc198552777)

[2. OVERALL DESCRIPTION 4](#_Toc198552778)

[2.1 Product Perspective 4](#_Toc198552779)

[2.2 Product Functions 4](#_Toc198552780)

[2.3 User Needs 5](#_Toc198552781)

[2.3 User Characteristics 5](#_Toc198552782)

[2.3 Assumptions & Constraints 5](#_Toc198552783)

[Assumptions: 5](#_Toc198552784)

[- Constraints: 5](#_Toc198552785)

[3. SPECIFIC REQUIREMENTS 6](#_Toc198552786)

[3.1 Functional Requirements 6](#_Toc198552787)

[3.1.1 User Management & Role Selection 6](#_Toc198552788)

[3.1.2 Upload and Categorization 6](#_Toc198552789)

[3.1.3 Search and Browse 6](#_Toc198552790)

[3.1.4 File Management 6](#_Toc198552791)

[3.1.5 Role-Based Access Control 7](#_Toc198552792)

[3.2 External Interfaces 7](#_Toc198552793)

[3.3 Use Case Descriptions 7](#_Toc198552794)

[Use Case ID: UC-1 7](#_Toc198552795)

[Use Case ID: UC-2 8](#_Toc198552796)

[3.4 Performance Requirements 8](#_Toc198552797)

[3.5 Logical Database Requirements 8](#_Toc198552798)

[3.6 Design Constraints 9](#_Toc198552799)

[3.7 Software Quality Attributes 9](#_Toc198552800)

[3.8. Non-Functional Requirements 9](#_Toc198552801)

[3.8.1 Performance 9](#_Toc198552802)

[3.8.2 Security 10](#_Toc198552803)

[3.8.3 Usability 10](#_Toc198552804)

[3.8.4 Compatibility 10](#_Toc198552805)

[4. Object-Oriented Models 10](#_Toc198552806)

[5. System Architecture (High-Level) 12](#_Toc198552807)

[5.1 Components 12](#_Toc198552808)

[5.2 Data Flow 13](#_Toc198552809)

[6. Future Enhancements 13](#_Toc198552810)

[7. Conclusion 13](#_Toc198552811)

[8. APPENDICES 13](#_Toc198552812)

[8.1 Glossary 13](#_Toc198552813)

# 1. INTRODUCTION

## 1.1 Purpose

This document specifies the software requirements for the development of the Past Question Repository System (PQRS) at Nile University of Nigeria. The purpose is to define the system that will serve as a centralized, searchable, and secure database where students can access past exam questions, while Course Representatives and Advisors manage and organize content.

The intended audience includes developers, testers, project managers, stakeholders, and university administrators involved in the PQRS system.

## 1.2 Scope

The MVP system will allow:

• Students to browse, search by course code, and download past exam questions.

• Course Representatives to upload and categorize PDF question papers.

• Basic role selection (Student/Course Rep) without full authentication.

• Simple file management with PDF format support only.

The system will not include course registration, academic records, or examination scheduling functionalities.

## 1.3 Definitions, Acronyms, and Abbreviations

• PQRS : Past Question Repository System

• NUN : Nile University of Nigeria

• CR : Course Representative (Course Reps)

• CA : Course Advisor

• UI : User Interface

• API : Application Programming Interface

• PDF : Portable Document Format

• DOCX : Microsoft Word Document Format

• JPEG/PNG : Image Formats

## 1.4 Overview

The PQRS aims to provide an efficient digital solution for managing and accessing past exam questions. It will reduce reliance on physical archives, improve accessibility, and ensure proper categorization and management of academic resources.

# 2. OVERALL DESCRIPTION

## 2.1 Product Perspective

The PQRS will function as a standalone application with integration capabilities for NUN's existing student information system and authentication infrastructure. It will support web and mobile access.

## 2.2 Product Functions

Key functions include:

• User Authentication

• Upload and Categorization of Past Questions

• Search and Browse Functionality

• Download Options

• Role-Based Access Control

• File Format Support

## 2.3 User Needs

• Students need easy access to past questions for exam preparation.

• Course Reps need a simple way to upload and categorize questions.

• Course Advisors must verify and approve uploaded content.

## 2.3 User Characteristics

• Students : Varying levels of technical proficiency; need intuitive UI for browsing/searching/downloading.

• Course Reps & Advisors : Moderate technical skills; responsible for uploading and organizing content.

• Administrators : Advanced technical knowledge; oversee system performance and manage user roles.

## 2.3 Assumptions & Constraints

### Assumptions:

• Users have internet access.

• Course Reps/Advisors are trained on upload procedures.

• Accurate course and student data is available from NUN's existing systems.

• Faculty members will cooperate in providing official past questions.

### - Constraints:

• Storage limitations (e.g., max file size per upload).

• Only approved files are visible to students.

• Must comply with NUN data privacy and security policies.

• Must be compatible with common browsers and devices.

• Database must support scalability and backup mechanisms.

# 3. SPECIFIC REQUIREMENTS

## 3.1 Functional Requirements

### 3.1.1 User Management & Role Selection

• REQ-1 : Users select role (Student/Course Rep) without authentication.

• REQ-2 : Basic role-based access control.

• REQ-3 : Session management for role persistence.

### 3.1.2 Upload and Categorization

• REQ-4 : Course Reps can upload PDF past questions.

• REQ-5 : Files are categorized by course code and year.

• REQ-6 : Basic metadata (upload date, uploader).

• REQ-7 : Simple file preview before upload.

### 3.1.3 Search and Browse

• REQ-8 : Students can search by course code, year, or keywords.

• REQ-9 : Filter options by semester, exam type, or lecturer.

• REQ-10 : Display results with preview snippets and metadata.

### 3.1.4 File Management

• REQ-11 : Support for PDF, DOCX, JPEG, PNG.

• REQ-12 : Version control for updated question papers.

### 3.1.5 Role-Based Access Control

• REQ-13 : Students can only view/download.

• REQ-14 : CRs/CAs can upload, edit, delete within their assigned department.

• REQ-15 : Admins can manage all users and content.

### 3.2 External Interfaces

• Authentication : Integrated with NUN SSO system.

• Email Server : For notifications and password resets.

• Storage API : Cloud storage for file persistence (e.g., AWS S3 or Google Cloud Storage).

## 3.3 Use Case Descriptions

### Use Case ID: UC-1

• Name : Student Downloads Past Question

• Actors : Student

• Goal : Download past exam questions

• Trigger : Student wants to prepare for exams

• Precondition : Selected Student role

• Postcondition : PDF file downloaded

#### Basic Flow :

• Student enters course code in search.

• Selects question from results list.

• Clicks "Download" button.

• PDF file is downloaded.

### Use Case ID: UC-2

• Name : Course Rep Uploads Question

• Actors : Course Rep

• Goal : Upload PDF past question

• Trigger : New question available

• Precondition : Selected Course Rep role

• Postcondition : File uploaded with metadata

#### Basic Flow :

• Selects "Upload" option.

• Enters course code and year.

• Uploads PDF file.

• Previews and confirms upload.

## 3.4 Performance Requirements

• System should handle at least 500 concurrent users.

• Search queries should return results in under 2 seconds.

• Upload/download speed should be optimized for large files.

## 3.5 Logical Database Requirements

The system will use a relational database to store:

• Users : ID, name, email, role, password hash.

• Courses : Code, title, department.

• Past Questions : File path, metadata, upload date, uploader ID.

• Tags : Keywords for search optimization.

• Access Logs : For audit purposes.

## 3.6 Design Constraints

• UI must follow responsive design principles.

• Must adhere to NUN branding and accessibility standards.

• All file transfers must be encrypted.

## 3.7 Software Quality Attributes

ATTRIBUTE DESCRIPTION

Usability Intuitive navigation, minimal learning curve

Security Encrypted storage, role-based access

Performance Fast response times even during peak usage

Scalability Capable of supporting increased load over time

Reliability Minimal downtime, regular backups

Maintainability Modular design for easy updates

Compatibility Cross-browser and cross-device support

Accessibility WCAG-compliant interface

## 3.8. Non-Functional Requirements

### 3.8.1 Performance

• The system should handle \*\*100+ concurrent users\*\* without slowdowns.

• File uploads should process in \*\*<5 seconds\*\* (for files up to 20MB).

### 3.8.2 Security

• Role-based access control (RBAC): to restrict unauthorized actions.

• Data encryption: for file storage and user credentials.

### 3.8.3 Usability

• Mobile-friendly: interface for students accessing on phones.

• Intuitive UI: for easy uploads and searches.

### 3.8.4 Compatibility

• Supports: Windows, macOS, Android, and iOS.

• Accepts: PDF, DOCX, JPEG, PNG ( file formats).

# 4. Object-Oriented Models

+-------------------+

| User |

+-------------------+

| - id |

| - name |

| - email |

| - role |

| - passwordHash |

+-------------------+

| + login() |

| + logout() |

+-------------------+

+-----------------------+

| PastQuestion |

+-----------------------+

| - fileId |

| - courseCode |

| - year |

| - semester |

| - examType |

| - filePath |

| - uploaderId |

| - uploadDate |

+-----------------------+

| + upload() |

| + download() |

+-----------------------+

+----------------------+

| CourseRep |

+----------------------+

| - assignedCourses[] |

+----------------------+

| + uploadQuestion() |

| + editQuestion() |

+----------------------+

+----------------------+

| CourseAdvisor |

+----------------------+

| - assignedCourses[] |

+----------------------+

| + approveUpload() |

| + deleteQuestion() |

+----------------------+

# 5. System Architecture (High-Level)

## 5.1 Components

• Frontend: Flutter Web & Mobile App.

• Backend: Python Flask REST API.

• Database: SQLite (stores metadata).

• File Storage: Local file system for MVP phase.

## 5.2 Data Flow

• Course Rep: uploads PDF file with metadata.

• System: stores file and indexes metadata.

• Students: search and download files directly.

# 6. Future Enhancements

• AI-based search (e.g., OCR for handwritten past questions).

• Discussion forums: for students to share solutions.

• Automated notifications: for new uploads.

# 7. Conclusion

This SRS defines the requirements for a \*\*Past Question Repository System\*\* that ensures secure, efficient access to past exam materials while maintaining content quality through an approval workflow.

# 8. APPENDICES

## 8.1 Glossary

• Past Question : A previous exam paper made available for revision.

• Course Rep : A student representative for a particular course.

• Course Advisor : A faculty member who supervises course-related activities.

• Repository : Centralized storage location for digital resources.

• Metadata : Data describing other data (e.g., author, date, category).